**Census Income**



In ancient times, the ability to predict the future was called precognition. Nowadays we call it machine learning. I recently stumbled upon a data which was quite interesting to work on as it gives a picture on if any persons average income stays above 50K or below that.

Though the data is pretty old, but that will still increase our knowledge to work with data. I am not actually sure if that works for current work structure that we have, but let's see how it goes.

This data was extracted from the 1994 Census bureau database by Ronny Kohavi and Barry Becker (Data Mining and Visualization, Silicon Graphics). A set of reasonably clean records was extracted using the following conditions: ((AAGE>16) && (AGI>100) && (AFNLWGT>1) && (HRSWK>0)).

The idea behind the workshop is to ingest data from a website, perform some initial analysis to get a sense for what’s in the data, then structure the data to fit a Scikit-Learn model and evaluate the results.

The dataset contains information about the annual incomes of people from 42 different countries, but the majority (90%) is dominated by the United States. The runner-up in this category is Mexico at 2%, leaving only 8% for the other 40 countries.

**Motivation:**

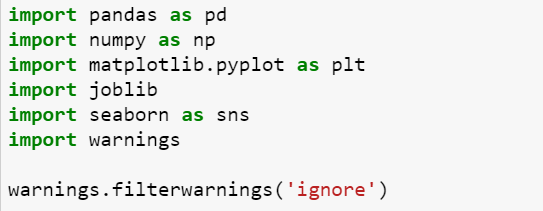
The main objective behind our project is to predict if a said person, given his attributes, earns more than $50k per annum or not.

We begin by cleaning the data, and then move-on to exploratory data analysis of the dataset. Following that, we prepare the data for our machine learning model and then train the model using that data.

I know that it was simple to hear, but let's make it simple and start digging into the data.

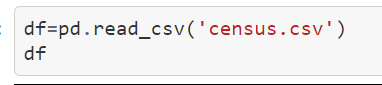
Let us get started!!!...

The initial step that we all know will of course be importing important data, I believe it is not necessary to include all the importing file at the top, so let us just import them whenever needed.



These are some basic files which I believe is necessary to do initial check in the data.

Using pd.read\_csv we can get the data in any data(df here.)

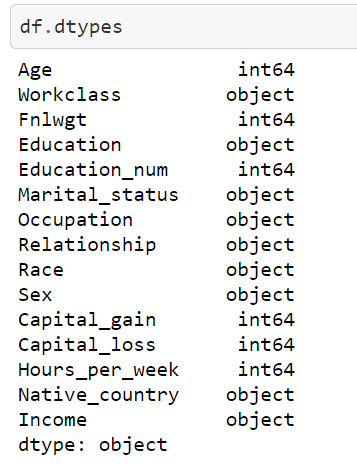


By glancing at the first 5 rows of the data, we can see that we have primarily categorical data.it would be great to see the frequencies of each class. To do this, we can use Seaborn's count plot function to count the occurrences of each data point and also to get the details of the columns comparing it with the income.

The dataset contains 32,560 entries with a total of 15 columns representing different attributes of the people.

Initial step to move on will be the data cleaning as its best keep your data tidy.

We know that there are columns mostly with categorical variables but we have to get the details on which are the columns which are under object or in others. Luckily pandas gives us a leverage on that by having a function 'dtype' to get the columns segregated based on the data type.



Now as we have an understanding on the categorical and the numerical part of the data, we will proceed further to check if there are any null values in the data.

After having a check for the null values, we notice that there are no null values, lucky right...

but there is something else incoming.

**Pre-processing:**

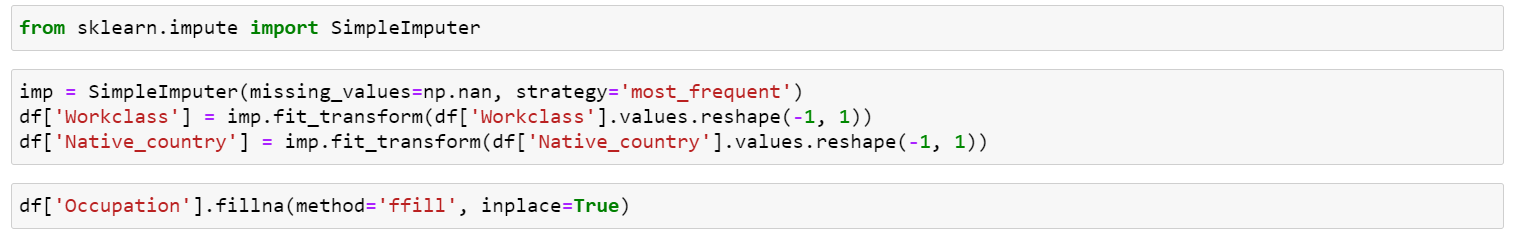
On a detailed check, we see that there is ‘?’ for missing values which we have to find and replace.

Initially, we can use the replace command to substitute the values to Nan and thereafter we have a very useful imputer from sklearn which helps us to impute the nan values depending on the column.

We have values in Workclass, Native country and Occupation.

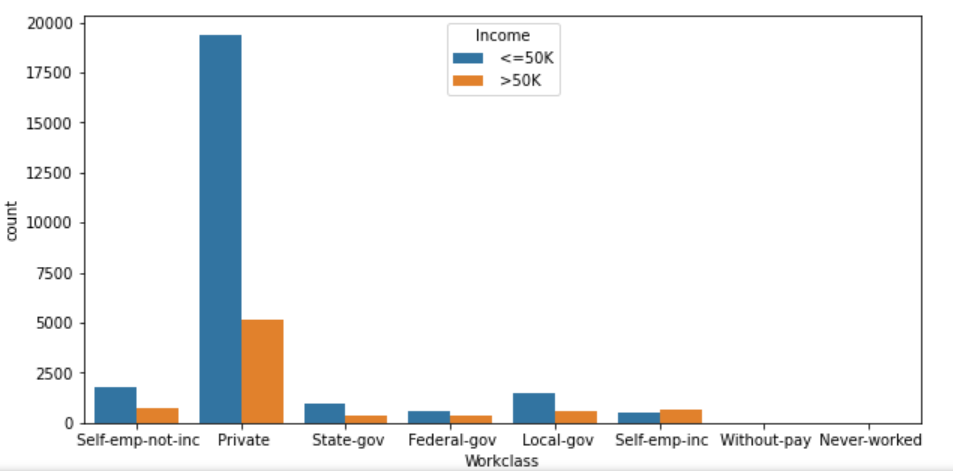
So, we can use the mode to substitute ‘?’ values in Workclass and native country and ffill to fill the Occupation.

For those who dont know what ffil is, ffill() function is used to fill the missing value in the dataframe. 'ffill' stands for 'forward fill' and will propagate last valid observation forward. inplace : If True, fill in place.

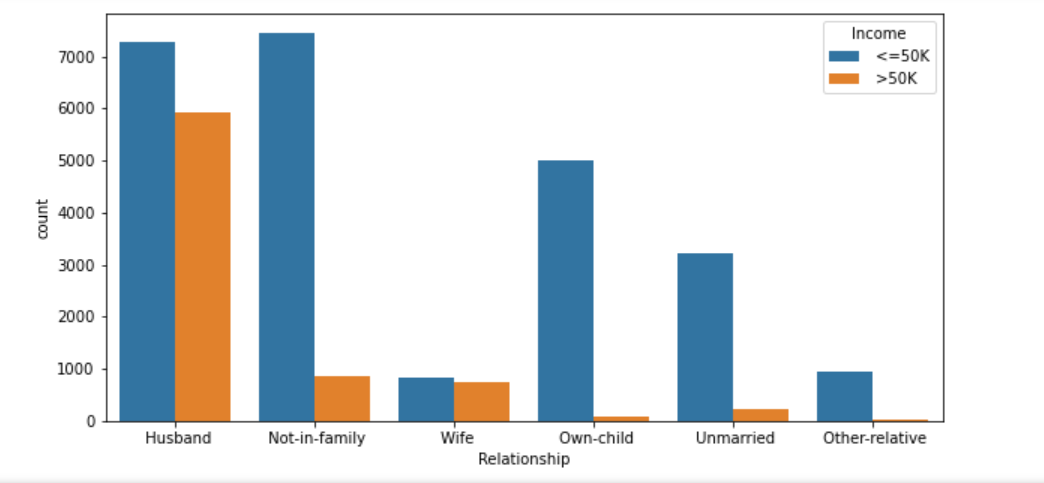


So, the cleaning part is done and we can check for the relevance of the data here.

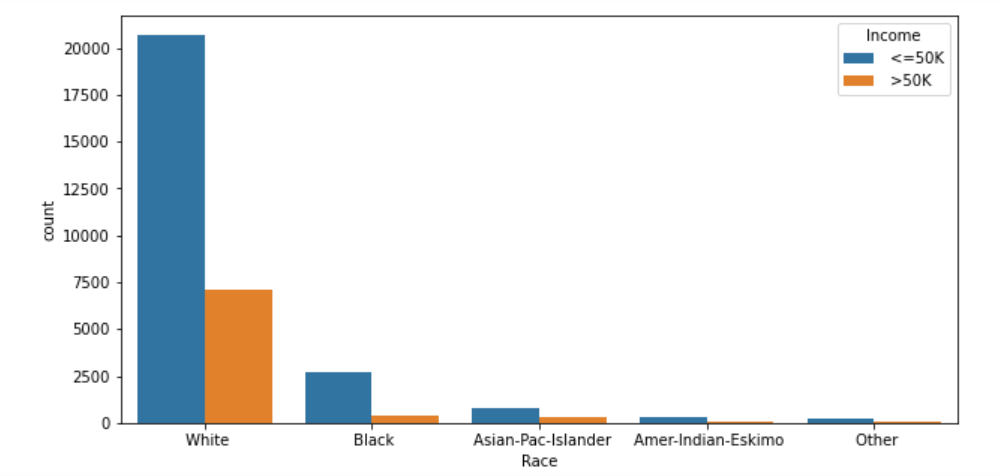
**Exploratory data analysis:**



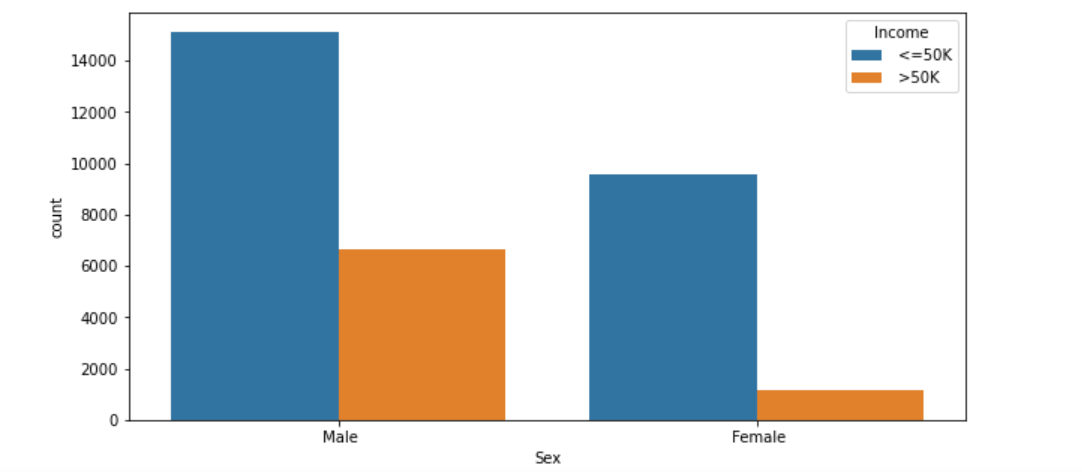
Now that was as expected, People in Private sector tend to get more salary than the other working class. Looks like we have to start our career in private sectors to earn more..lol!



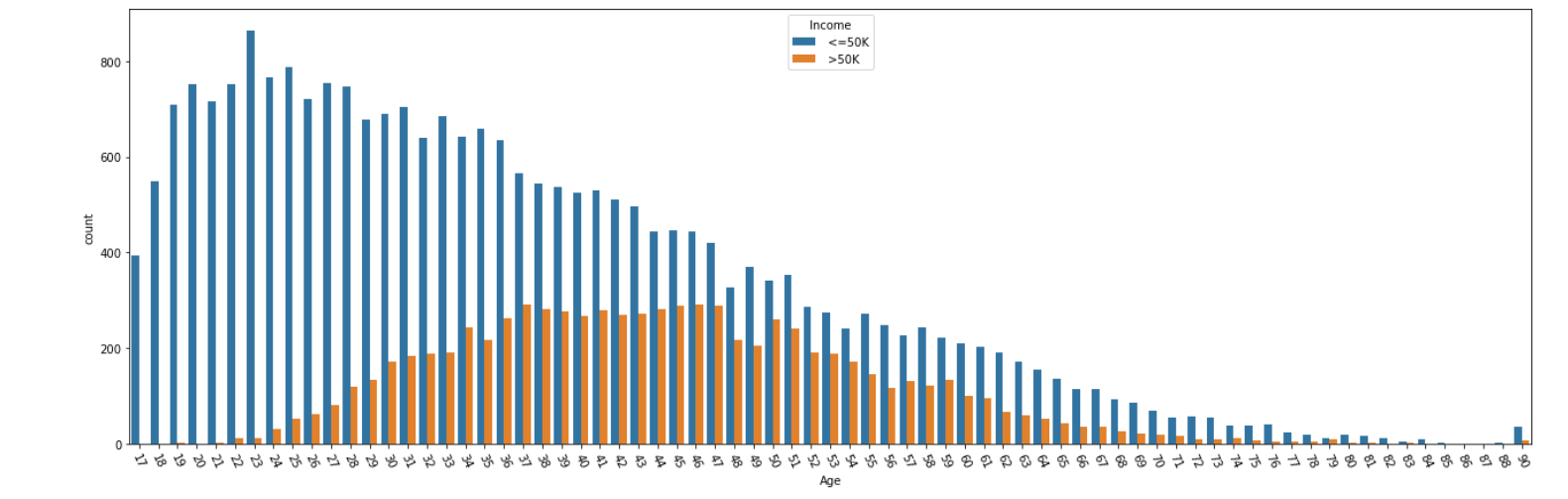
We see that there is not much change in the salary for husbands but when we look for people who are not in family, very less tend to get salary more than 50K, same with others as well.



Looking at this we can conclude that the whites are in more number and most of them get a salary less than 50K.

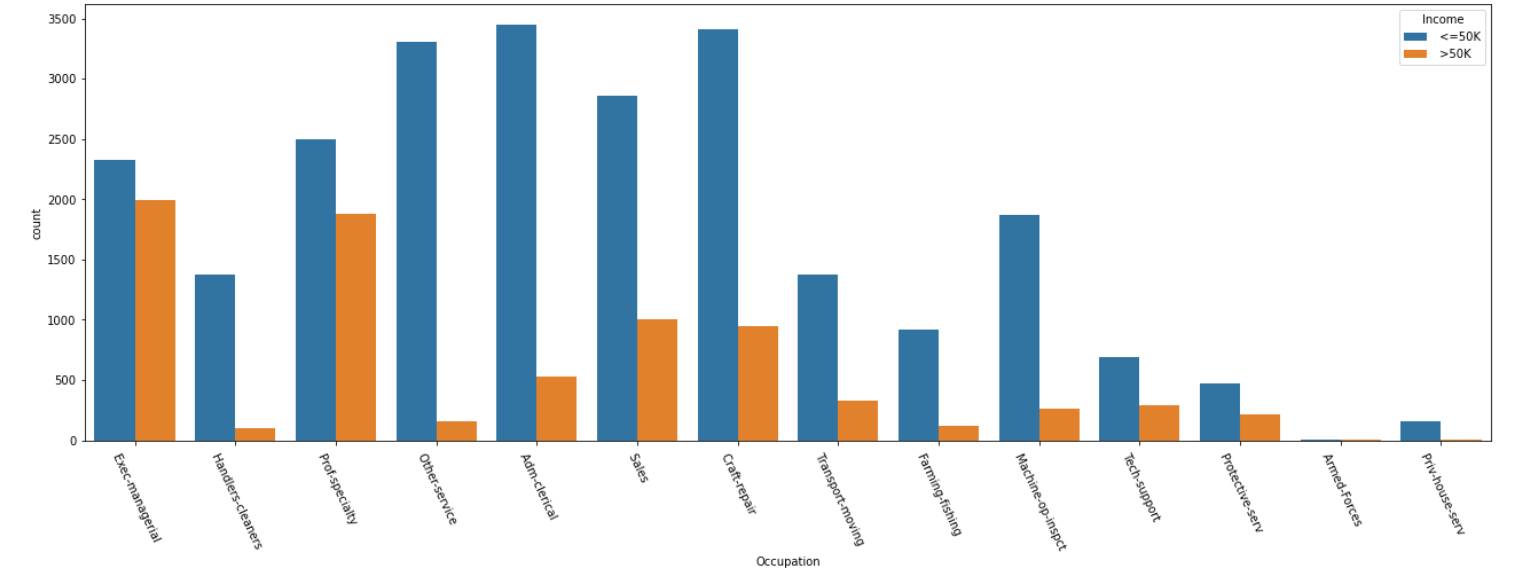


More male tends to take more than 50K that compared to their female counterpart.

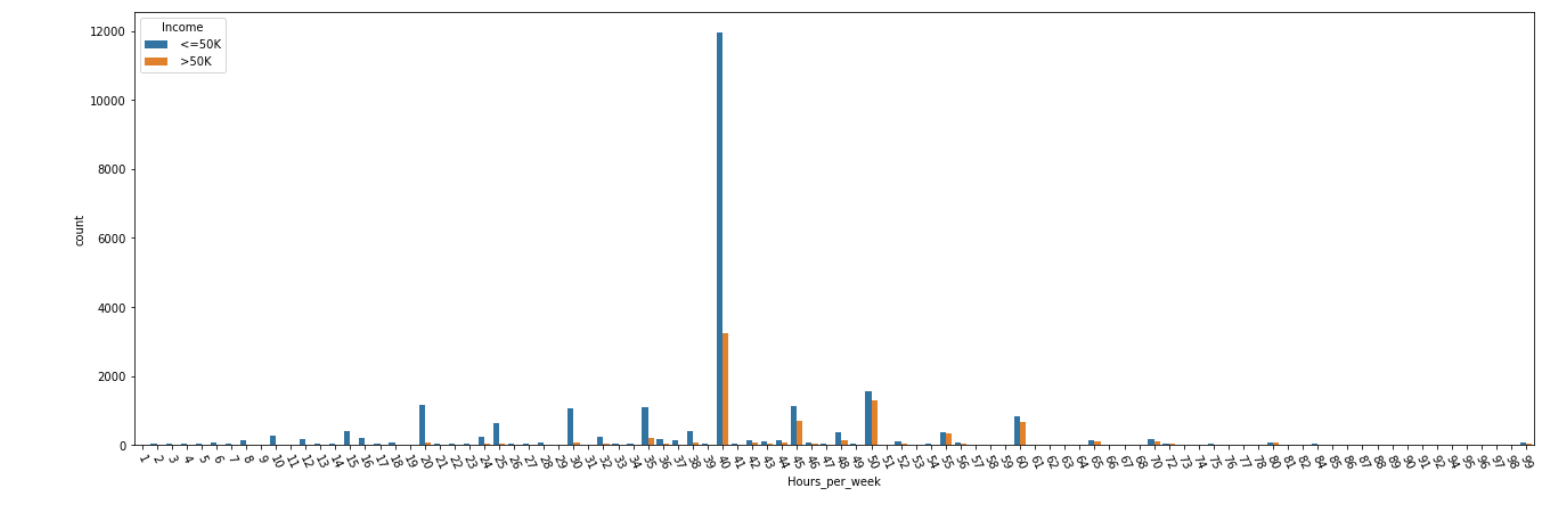


We see here that the median of the people earning less than 50K is higher compared to the median of people earning more than 50K.

With the increase in age, we see that more people tend to get salary higher than 50K.



Looks like exec managerial post and prof secretary might fetch you a hefty salary. Start working to get that post..!!



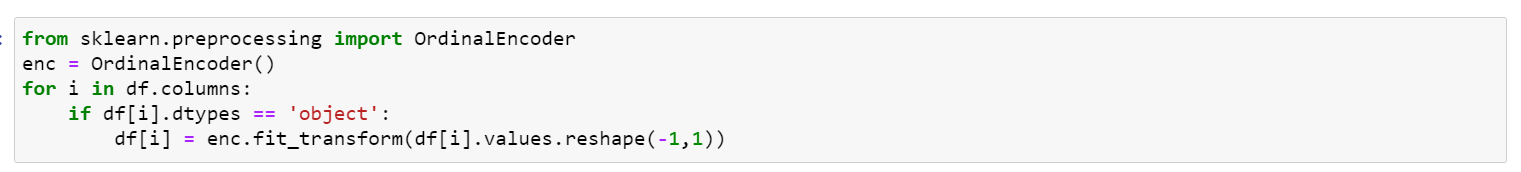
We see here that most of the people are working for 40 hours per week and there are people who earn more than 50K in that category. and with the increasing the number of hours, there is a possibility that you get more than 50K and eventually if you work too much, not good for you always..ha ha..

As we have got some very needed insights here, we can work to encode the data and thereafter check the correlation of the plot.

Our first step is to get our data out of the object data type land and into a numeric type, since nearly all operations we’d like to apply to our data are going to rely on numeric types.

We have a very useful encoder here which is ordinal encoder which treats every column individually and encodes it to avoid machine to consider every encoded value to be the same.

If we use label encoder, it will not consider same values in different column as separate, it will consider that to be the same which can yield a poor result afterwards.



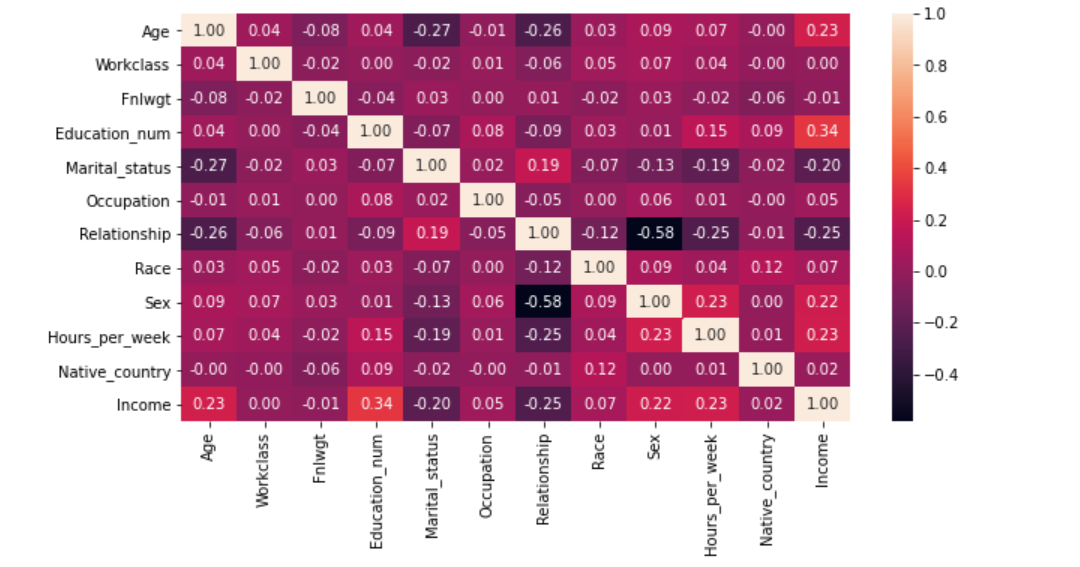
This encodes all the object variables. Always better to get everything in one go rather than to encode each and every row.

Let us now check the correlation plot here.

First thing first, what is correlation:

Correlation coefficients quantify the association between variables or features of a dataset.

Which means we get to know what are the factors which affect the data compared to each other. The correlation coefficient is determined by dividing the covariance by the product of the two variables standard deviations.



The observations here are:

1. There is a positive relation between the Age and income which means with the increment in age, there is a change that people might get higher salaries.

2. The marital status and relationship are positively correlated to each other whereas both are negatively correlated with income.

3. Hours per week has a positive correlation with the income.

4. Income and education is positively correlated.

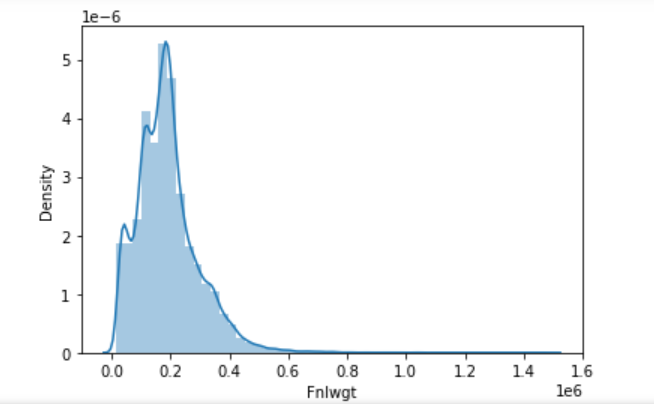
Now these data make more sense as a person working more hours tends to get higher salary and a person with more education earns more.

Uff, done with the EDA, that was fun.

I guess that was fun for all of you...

Opps, forgot to check the skewness here...

When we analyse the data using df,skew, in the numerical variables there will only be fnlwgt which has outlier and we can simply use the power transformer to remove it.



See...

To get this curve normalised, we can use the yeo Johnson method, that will make sure that the curve is normalised, also we can try using the log transform as well because of the positiveness in the data, there are no negative values here.

swissh..thats done!!

Now it’s better to split the data to x and y.

And yes, I have also dropped the capital gain and loss and this do not infer much data here.

And looking at the data that we have its always best to scale the data so that we can get the whole data into one scale.



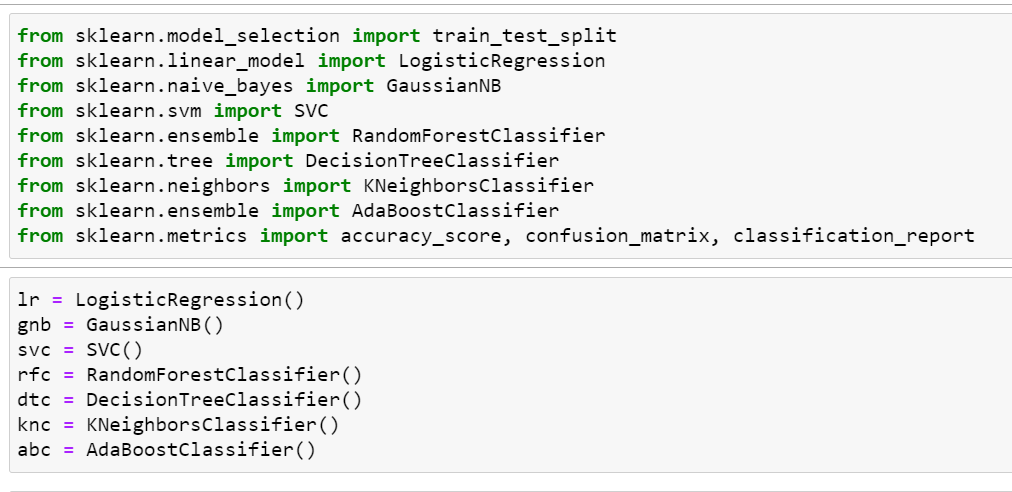
Thinking what is x\_col.. that is just the column names that I had taken so that I can replace at this step. Scaling might change the data to a 1D array and using the pandas we can convert them to data frame and assign columns names to them.

This was the time where I simply went through the whole steps done so far to make sure that the data is clean and there is nothing else, we can do to improve.

One thing that we have noted was that we have dropped capital gain and loss. I had removed that because of the 0 values in it and as it was a discrete variable.

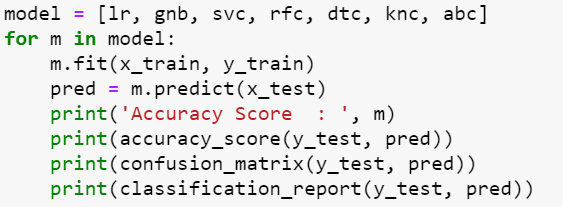
We can somehow work with that as well and do some feature engineering to get most out of it. But let’s keep t for some other day.

**Modelling:**

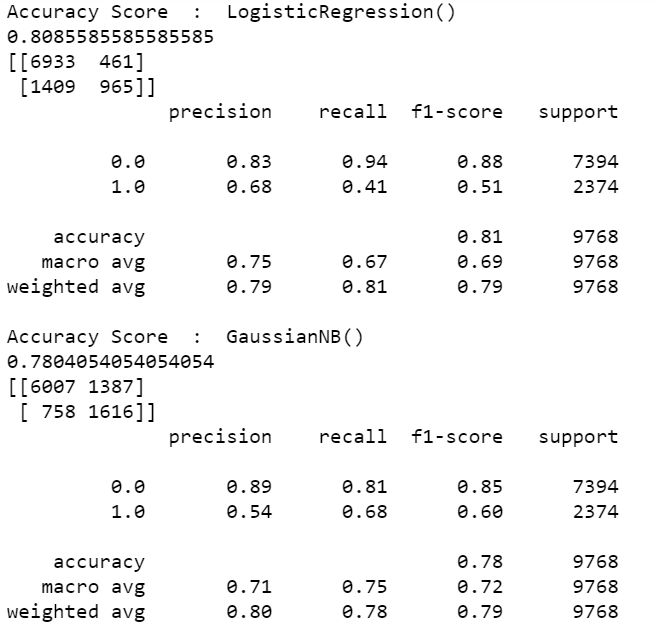


At the top as I was saying I don't like to import all at the top, I have imported the metrics and the model.

Just to make sure that we test every model and the accuracy, I will use a for loop here.



This will list out all the data with the accuracy and the classification report of the data.

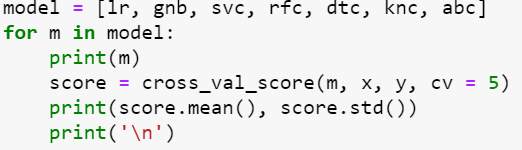


Something like this..

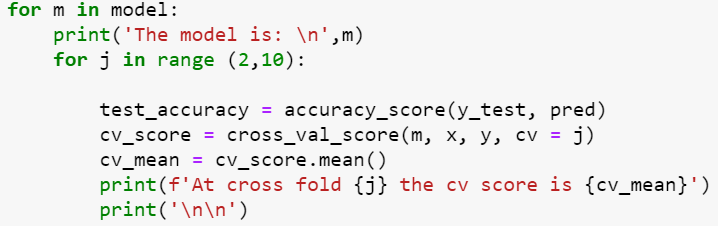
Reducing the overfitting/underfitting:

Your model is underfitting the training data when the model performs poorly on the training data. ... Your model is overfitting your training data when you see that the model performs well on the training data but does not perform well on the evaluation data.

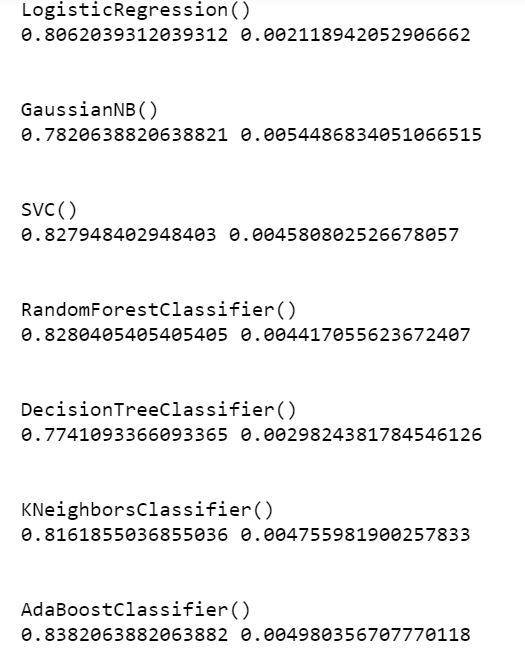
So, we can use the Cross validation to get most of the data fit.



We will get the result of all models with cv=5 here, we can also use a for loop here like this:



For this model we are considering the cv value to be 5 and the model will show the data like this:



That is the mean and standard deviation respectively.

We have found now that KNN is the best working model for me, though you guys can try with other models as well to see which is the best one working for you.

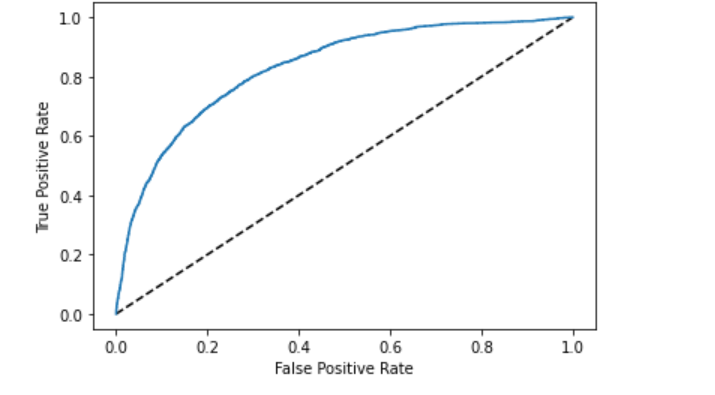
ROC AUC curve:

This is the most important part of the machine learning model that we have built as it visualises on how best the model is working in predicting the correct values.

AUC - ROC curve is a performance measurement for the classification problems at various threshold settings.

Higher the AUC, the better the model is at predicting 0s as 0s and 1s as 1s.

Here is mine:

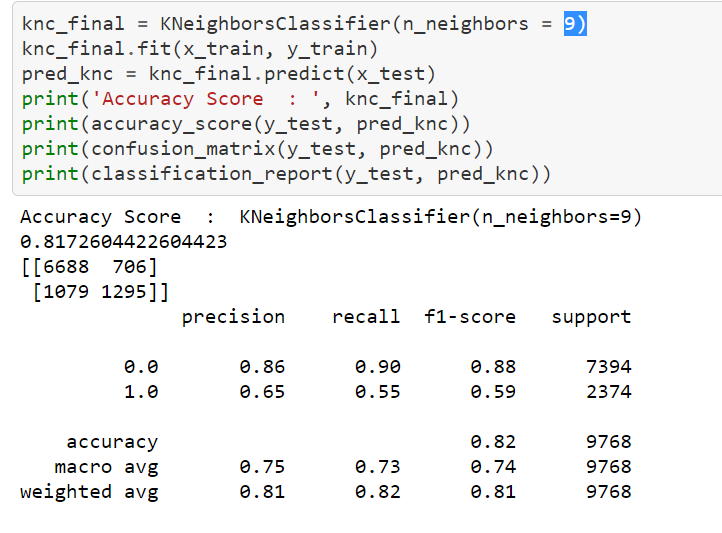


Not bad I guess...

To increase the accuracy even further, we can go ahead with hyperparameter tuning and get the best results out of the KNN.

We can set the parameters and use the Gridsearch CV to see if the model works at its best or not.

This is my final model and using the same to train the test data and finalising.



Looks good to me, what do you guys think?

Finally saving the model:

import joblib

joblib.dump(knc\_final, 'Census\_Incomekn.obj')

That’s the command and your file is ready to go!!

Conclusion:

With the provided attributes we got know the factors which determined the salary of a person and also the factors which decides if a person gets a higher or a lower salary.

We got a better accuracy to predict the average income of a person. Atleast looking at how people make their predictions based on limited knowledge, its sometimes better to go with what data shows us.

This was an end to end look at how I performed a classification analysis on the data, I had tried my best to include every possible details of the classification here to understand how the factors affect the income of a person.

Please let me know if you see any scope of improvement and I will be glad to look into that.

Thanks.